# Optimizing Trajectories Clustering for Activity Recognition

Guido Pusiol, Luis Patino, Francois Bremond, Monique Thonnant, and Sundaram Suresh

Inria, Sophia Antipolis,
2004 Route des Lucioles, 06902 Sophia Antipolis CEDEX, France
{Guido.Pusiol,Jose-Luis.Patino_vilchis,Francois.Bremond,Monique.Thonnat,
Sundaram.Suresh}@sophia.inria.fr
http://www-sop.inria.fr/pulsar

**Abstract.** This work aims at recognizing activities from large video datasets, using the object trajectories as the activity descriptors. We make usage of a compact structure based on 6 features to represent trajectories. This structure allows us to apply standard techniques for unsupervised clustering. We present a method to optimize trajectory clustering by tuning the set of trajectory feature weights in order to improve the performance of activity recognition. We have learned the weights and test the approach using different sets of real video data achieving domain knowledge independence. Moreover we define new performance measures that better describes the evaluation of the clustering process comparing the learned clusters with activity ground truth.

**Key words:** Behavior recognition, video surveillance, data mining, unsupervised trajectory clustering, performance evaluation

## 1 Introduction

Nowadays, the technical and scientific progress require human operators to handle large quantities of data. It becomes almost an impossible task to continually monitor these data sources manually. The data-mining field can provide adequate solutions to synthesize, analyze and extract information. Because of the advance made in the field of object detection and tracking [2] data-mining techniques can be applied on large video data. Data-mining on video data has mainly been employed for annotation/retrieval processes[4][5]. The task consists in classifying multiple video features in categories associated with meaningful semantic keywords that will allow the retrieval of the video. Usually low level features (i.e., color, texture, shape, and motion) are employed. A recent review in video retrieval can be found in [6]. Recently particular attention has been turned to the object trajectory information over time to understand high level behaviors. Pioneering work by Johnson and Hogg [7] have presented a learning behavior technique using neural networks. Owens and Hunter [8] have used a Self Organizing Feature Map to learn behaviors. Trajectories have been shown

to be useful on their own for activity clustering. For instance Piciardelli et al. [12] employ a splitting algorithm applied on very structured scenes (such as roads) represented as a zone hierarchy difficult to generalize on other domains. Anjum et al.[13] employ PCA to reduce the dimensionality of trajectories. They analyse the PCA first two components of each trajectory together with their associated average velocity vector. Mean-shift, with these features, is employed to seek the local modes and generate the clusters. Similarly, Naftel et al. [14] first reduce the dimensionality of the trajectory data employing Discrete Fourier Transform (DFT) coefficients and then apply a self-organizing map (SOM) clustering algorithm to find normal behavior. Antonini et al. [15] transform the trajectory data employing Independent Component Analysis (ICA), while the final clusters are found employing an agglomerative hierarchical approach. Calderara et al. [16] employ a kmedoids clustering algorithms on a transformed space modeling different possible trajectory directions to find groups of normal behavior. Abnormal behavior is detected as a trajectory that does not fit into the established groups but the approach is validated with acted abnormal trajectory. Gaffney et al. [17] employed mixtures of regression models to cluster hand movements, although the trajectories were constrained to have the same length. Hidden Markov Models (HMM) have also been employed [19] [18]. Most of these approaches have many parameters to be fixed, but none of them proposed solutions on how to tune them to improve the global performance of trajectory clustering.
This work extends a previous approach [10] for learning behaviors, where simple trajectory descriptors are used with the ability of managing huge amounts of data in a minimum computing time. This work focuses on exploring new techniques to tune the parameters of the clustering process and to evaluate the obtained clusters. We consider human activities (such as loitering, queuing at the ticket vending machine, passing the gates) which can be characterized by people trajectories.

## 2 Proposed Approach

### 2.1 Approach Overview

The monitoring system is mainly composed of two different processing components. The first one is an analysis subsystem for the real time detection of objects and events, is based in detection and tracking and is the responsible of producing the trajectories from the real data. The second one is an off line subsystem that aims at clustering the trajectories to extract the main behaviors occurring in the observed scene.

### 2.2 Online Subsystem

Online subsystem is composed mostly of two tasks : detection and tracking of mobile objects evolving in the scene. The detection of objects is performed using a thresholding operation between the pixel intensity of a frame with the pixel intensity of the background frame. The result is a binary mask of foreground pixels. The neighboring foreground pixels are grouped to form regions often referred to as "blobs". Using calibrated cameras we are able to calculate the 3D information (i.e., width and height) of moving objects as well as their 3D location on the ground plane. The 3D object information is compared against 3D model of an object (e.g., human, luggage, baggage, group or crowd). The "dynamic occlusion" problem is handled by classifying merged human objects in different categories (i.e., two merged human objects are considered to be a single group object and more merged humans are classified a crowd).
Our tracking algorithm [1] builds a temporal graph of connected objects over time to cope with the problems encountered during tracking. The detected objects are connected between each pair of successive frames by a frame to frame (F2F) tracker. Associations between objects are computed with a matching process based on three criteria: the similitude between semantic classes, 3D dimension and differences on 3D distance on the ground plane.
The graph of linked objects provided by the F2F tracker is then analyzed by the tracking algorithm, which builds paths of each mobiles according to the link features. The best path is then taken out as the trajectory of the related mobiles which is updated throughout the video.

### 2.3 Trajectory Representation

For the trajectory characterization we have selected a compact, comprehensive and flexible representation, that is suitable for large database analysis as opposed to many video systems which store the sequence of object locations for each frame of the video.
Each trajectory is defined by using six features: the entry point *(entry)* defines the beginning of the trajectory, exit point *(exit)* is the end of the trajectory, the angle from entry to exit points *(angle)*, the sum of the distance between all subsequent pairs of points *(walkeddist)*, the distance from *entry* to *exit* is

*(dist_ee)*, the mean variation of the angle between all subsequent pairs of points *(anglevar)*.
For example the angle variation of $t$ is defined as:
Let $T$ be the set of trajectories of a scene, for $t \in T$, $t_{i,x,y}$ represents the spatial coordinates of the $i^{th}$ point of $t$ and $|t|$ represents the number of points in the trajectory. Let angle $angle_{t_i}$ and $angle_{t_{i+1}}$ be the angles formed by the lines $(t_{i,x,y}, t_{i+1,x,y})$ and $(t_{i+1,x,y}, t_{i+2,x,y})$ with respect to the coordinate axes.

$$anglevar_t = \frac{\sum_{i=0}^{|t|-1} min(diff1, diff2)}{|t| - 1} \quad (1)$$

$$diff1 = max(angle_{t_i}, angle_{t_{i+1}}) - min(angle_{t_i}, angle_{t_{i+1}}) \quad (2)$$

$$diff2 = min(angle_{t_i}, angle_{t_{i+1}}) - max(angle_{t_i}, angle_{t_{i+1}}) + 360 \quad (3)$$

From now on the trajectory $t_i$ will be represented as a vector of features
$t_i =< f1_i, f2_i, f3_i, f4_i, f5_i, f6_i >$ were $f1_i = entry$, $f2_i = exit$, $f3_i = angle$, $f4_i = walkeddist$, $f5_i = dist_ee$, $f6_i = anglevar$.
We are aware that not all the features we have selected are independent. But this issue is handled by the importance (i.e., weights) given to each feature. The motivation behind the selection of this set of features is to capture meaningful information characterizing trajectories.

### 2.4 Filtering

Due to occlusion of objects and tracking failures, the set of extracted trajectories contains corrupted trajectories. Since the learning of the feature weights is done using real trajectories, we need to detect and filter errors introduced by the tracker. Here the problem is to differentiate between non frequent and erroneous trajectories.
To tackle this issue we eliminate from the set of trajectories only those with low probability of being a true trajectory. We consider that trajectories having 5 points or less of length, do not correspond to interesting behaviors (considering a frame rate of 5 fps). Also the case where $anglevar_t$ has abnormal patterns, we consider the tracking as erroneous, because many angle variations correspond in these cases to changes of tracking IDs.

### 2.5 Normalization

We have considered different methods to normalize the trajectory features based on their domain range, but finally we have adopted:
$\forall j \in T$:

$$< f1'_i, f2'_i, ..., f6'_i >=< f1_i - mean(f1_j), f2_i - mean(f2_j), ..., f6_i - mean(f6_j) > \quad (4)$$

$$tn_i =< f1'_i/std(f1'_j), f2'_i/std(f2'_j), ..., f6'_i/std(f6'_j) > \quad (5)$$

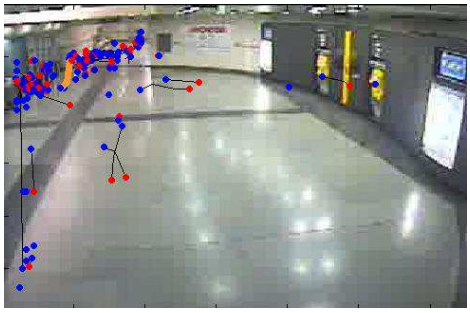

**Fig. 1.** Examples of filtered trajectories (red/blue dots are entry/exit points)

### 2.6 Feature weight

Each feature has a weight parameter associated at the clustering stage. The weight is a vector of 6 values, $W =< w_k >$ were $w_k$ is the importance of $f_k$ for all the trajectories.

### 2.7 Clustering

We feed the set of normalized and weighted feature vectors to a hierarchical clustering algorithm, some other clustering methods has been tested (i.e, SOM, K-means) [1] with worst performance. We employ the Euclidean distance as a measure of similarity to calculate the distance between all trajectory features. This simple technique allows a fast distance computation over a large amount of trajectories.

## 3 Clustering Evaluation

### 3.1 Ground Truth

To improve trajectory clustering by tuning feature weight and to evaluate the results, we have acquired behavior ground truths. A ground truth represents the frequent simple behaviors of people in the scene.
The scene is divided into interesting zones illustrated by figure 2 such as a vending machine, an elevator. These zones of interest are predefined by end users. Each behavior is represented by 3 trajectories that aims at capturing the expected paths between the interesting zones corresponding to a behavior.
The ground truth trajectories are represented using the same 6 feature vectors than the real data trajectories. We have modeled two ground truth sets corresponding to the two sites (i.e., Torino and Rome subway stations) where we have performed the experimentations. Torino and Rome ground truths are modeled using 100 and 50 expected behaviors respectively as shown in figure 3.
It is to be mentioned that two opposite behaviors such as: From “A” to “B” and From “B” to “A” have been modeled using their own set of independent trajectories.

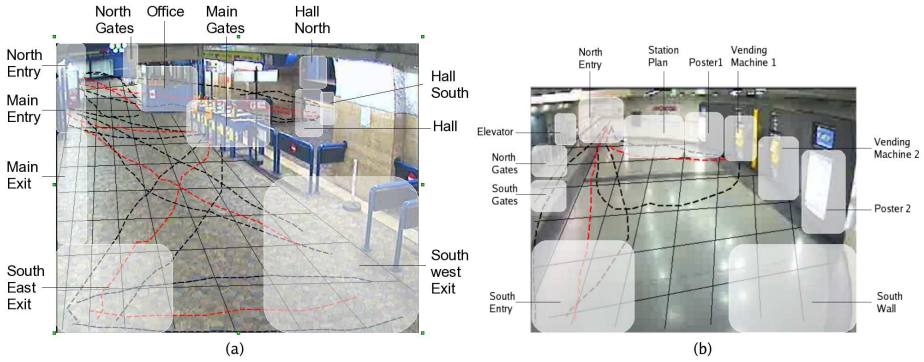


**Fig. 2.** (a) Rome Interesting Zones. (b) Torino Interesting Zones. Dashed lines represent examples of the ground truth trajectories associated to the behaviors predefined by end users.

### 3.2 Clusters Centers

A cluster center is a trajectory defined as the average of the trajectories features in a cluster, thus a center is a vector of 6 values. Each center is calculated in the domain of normalized and weighted features.
We use two types of cluster centers: the centers of real data clusters (RDC) and the centers of the ground truth clusters (GTC).
The GTCs are calculated by normalizing and weighting the ground truth trajectories associated with one behavior.
In figure 4 an illustration of GTC and RDC is provided. Graphically we only display the *entry* and *exit* features of the centers, which correspond to an approximation of the trajectory helping to understand the clustering performance.

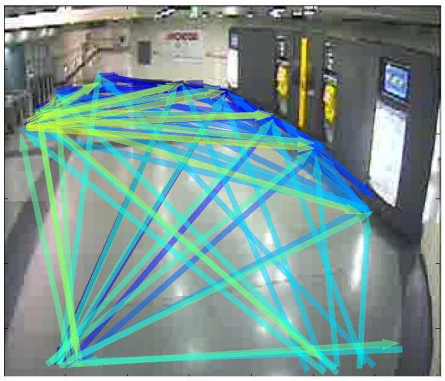

**Fig. 3.** The arrows correspond to Torino GTCs, representing 100 behaviors.

### 3.3 Performance Measures

To evaluate the clusters performance we have used two techniques. The first technique is based on clustering indexes such as Silhouette [3], which computes general quality information on the clusters in terms of density (small intra distance and big inter distance between the clusters). To calculate inter-intra cluster distances we use the normalized weighted features.

The second technique consists in defining performance measures based on ground truth in order to optimize the clustering performance with respect to end user requirements. We have defined four performance measures: *True Positive (TP), False Positive (FP), Dispersion, Quality.* TP is the number of correct associations between RDCs and GTCs. FP is the number of RDCs without any associated GTC. Dispersion measures how many RDCs are associated to a GTC. Quality indicates the mean distance between a GTC and its associated RDCs.

After clustering the real-data trajectories in a fixed number of clusters, the center of each cluster is calculated. Once we build the set of real-data clusters centers, the euclidean distance between each real-data center against all the ground-truth centers is estimated. We associate each real-data center to the cluster center whose distance is the minimal. The RDC's whose distance to a GTC is lower than a threshold distance $Thres$ are used to estimate *Dispersion.* We use a threshold to divide frequent and unfrequent or not modeled behaviors. Once each real-data cluster center is associated to a Ground-Truth center, situations like the showed in the figure 4 are going to be measured. In figure 4 left side, there are two real data centers associated with one Ground truth center. If we look carefully the trajectories of both clusters they are representing the same behavior, but because of tracker errors some of the trajectories are shorter than others. Informally *Dispersion* represents the amount of real-data clusters that appears splited when they are supposed to be merged in one. The improvement of this performance measure, using weighted features lead us to situations like the left side of figure 4 where two clusters have been merged.
Also the *Quality* of the RDCs accepted as a modeled behavior is measured by calculating the mean distance between them and the GTCs.

To calculate Dispersion, Quality, TP and FP, let $GT$ to be the set of clusters centers and $RD$ to be the real-data set of clusters centers. Let $g_i \in GT$ for $i \in 1..|GT|$ and $r_j \in RD$ for $j \in 1..|RD|$.
$\forall t \neq j$

$$(g_i, r_j) \in Associations\ iff\ (dist(g_i, r_j) < dist(g_t, r_j))\ \&\ (dist(g_i, r_j) < Thres) \quad (6)$$

$$(g_i, r_j) \in True\_Associations\ iff\ (g_i, r_j) \in Associations\ \&\ (dist(g_i, r_j) < dist(g_i, r_l)) \quad (7)$$

$$TP = |True\_Associations| \quad (8)$$

$$FP = |RD| - TP \quad (9)$$

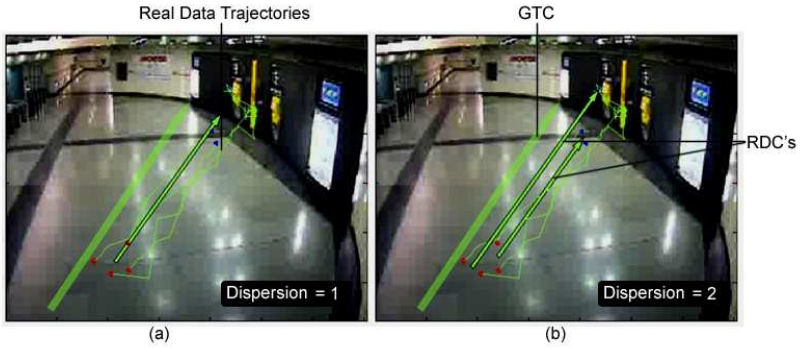


**Fig. 4.** Left: Clustering with weighted features, Right: Clustering without weights

$$Dispersion = \frac{\sum_{l=1}^{|GT|} \sum_{q=1}^{|RD|} |(g_l, r_q) \in Associations|}{TP} \tag{10}$$

$$Quality = \frac{\sum dist(a,b)}{|Associations|}, \quad \forall (a,b) \in Associations \tag{11}$$

### 3.4 Optimization Method

To optimize the subsystem for the extraction of meaningful activity patterns, two independent procedures are performed to compute the best number of trajectory clusters and to learn the feature weights.

First the optimum number of trajectory clusters for a scene is learned by an exhaustive search procedure through the cluster space. The quality of the cluster number is evaluated by calculating the Silhouette index. At this stage we do not assign any particular weight to the features used for clustering (i.e., $W =< 1, 1, 1, 1, 1, 1 >$).
Second the optimization of the feature weights is performed at clustering stage. This optimization process is divided in two stages:
The first stage (figure 5(a)) is to calculate the trajectory features for the real-data and ground-truth data. We normalize the real-data features with the method described in the previous section. The standard deviation and mean calculated from real data are the parameters used for normalizing the ground truth data.

The second stage (figure 5(b)) comprises the optimization of a performance measure (i.e., Dispersion) by changing the weights of the features. We use the cluster number obtained from Silhouette to fix this number at this stage. We use Genetic Algorithm (GA) [11] optimization technique to obtain the feature weights. At each iteration, GA determinates weights that is combined with the features of the normalized ground truth and real-data features vectors.

The Euclidean Distances between the weighted features are fed to the agglomerative clustering method, producing the set of real-data clusters. The RDCs and GTCs are calculated and *Dispersion* is measured from this sets. The obtained *Dispersion* is fed back to the GA method to be evaluated. GA stops after a fixed number of steps, returning the improved weight that minimizes *Dispersion* for a fixed number of clusters.

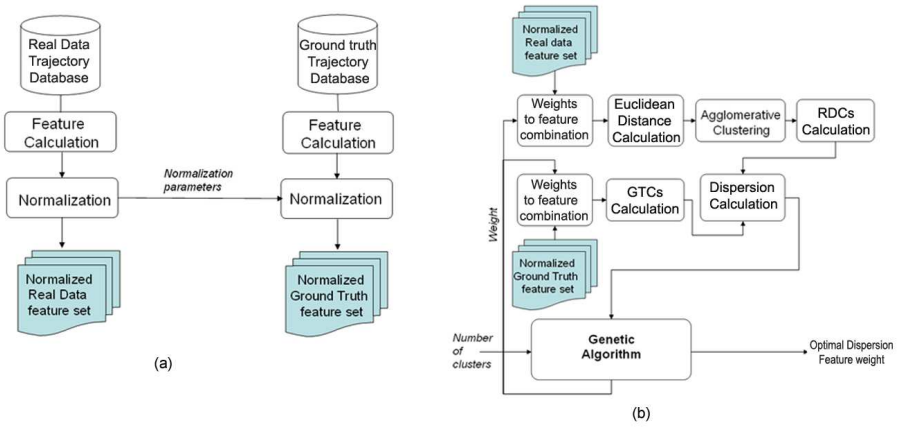


**Fig. 5.** (a) Feature Calculation and Normalization is performed in the same way for RDCs and GTCs (b) The optimization process contain 6 stages from weight assignment to feature up to GA algorithm

## 4 Experimental Results

### 4.1 Data Sets

We have used two dataset one in Rome and one in Torino subway stations. The Torino dataset contains 650 trajectories corresponding to 45 minutes. This dataset is used to learn the trajectory feature weights. The Rome dataset contains 5 hours of video, from were 5132 trajectories remain after filtering. The last is used to evaluate the whole process.

### 4.2 Computing the Cluster Number

To calculate a Silhouette index we had to explore the range of the possible cluster numbers. After observation of real trajectories, we have figured out that at least 18 different behaviors could be observed. Thus we have calculated Silhouette index variating the cluster number between 18 and 80. In figure 6 is displayed

the exhaustive search results, changing the cluster number. It can be noticed that 22, 32, and 70 are the cluster numbers maximizing the Silhouette index for the for Torino Scene. Since silhouette is only a general clustering index, we

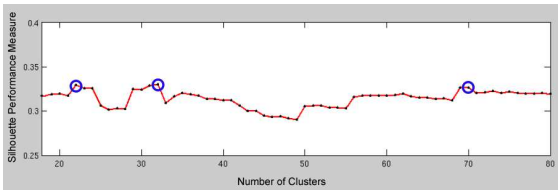


**Fig. 6.** Silhouette performance measure of Torino Dataset

cannot assert that the learned cluster number is the optimal one, however it is an approximation of were we should fix the cluster number to have a good description of different behaviors.

### 4.3 Learning the Feature Weights

We have performed the learning stage for several cluster numbers around 32 to check the stability of the learned weights, this way we assert that the GA algorithm did not find a "local minimum". The GA was set to perform 300 iterations for each cluster number, using a population size of 40 chromozomes. The feature weights can take values in [0..1] of real numbers. We have performed 4 runs of the optimization procedure variating the cluster number from 26 to 36 clusters. The average of the learned weights is displayed in the figure 7. A certain stability

| Number of clusters | w1 | w2 | w3 | w4 | w5 | w6 | Dispersion |
|---|---|---|---|---|---|---|---|
| 26 | 0.70 | 0.53 | 0.76 | 0.38 | 0.24 | 0.00 | 1 |
| 27 | 0.60 | 0.57 | 0.92 | 0.51 | 0.17 | 0.17 | 1 |
| 28 | 0.61 | 0.60 | 0.87 | 0.49 | 0.63 | 0.12 | 1 |
| 29 | 0.75 | 0.78 | 0.98 | 0.46 | 0.34 | 0 | 1 |
| 30 | 0.89 | 0.71 | 0.78 | 0.57 | 0.38 | 0.00 | 1 |
| 31 | 0.74 | 0.59 | 0.91 | 0.29 | 0.13 | 0.07 | 1 |
| 32 | 0.72 | 0.63 | 0.95 | 0.30 | 0.25 | 0.10 | 1.03 |
| 33 | 0.81 | 0.60 | 0.76 | 0.33 | 0.30 | 0.03 | 1 |
| 34 | 0.82 | 0.44 | 0.85 | 0.33 | 0.13 | 0.14 | 1.03 |
| 35 | 0.81 | 0.56 | 0.91 | 0.23 | 0.14 | 0.18 | 1.02 |
| 36 | 0.87 | 0.54 | 0.92 | 0.21 | 0.32 | 0 | 1.02 |

**Fig. 7.** Optimization procedure results

in the weight values can be noticed in the figure 7 table. The experiments show that *angle* (i.e., w3) is the most important feature, followed by the *entry* (i.e., w1) and *exit* (i.e., w2) points. We have chosen $W =< 0.75, 0.6, 0.9, 0.4, 0.3, 0.1 >$ as the learned weights. These weights are defined using the average of the weight

values presented in the figure 7.
We have found that since the number of clusters is a fixed value, the minimization of *Dispersion* not only enables to merge similar behaviors, but also helps to split clusters that do not correspond to the same behavior.

## 5 Evaluation

Our evaluation method follows a classical protocol. Having already learned the weights for clustering, the evaluation is performed by measuring clustering performances described in previous section using a completely different scene, dataset and ground truth. To validate the improvement, we have compared the performance measures between clustering with and without applying the learned weights. In this section we not only show the improvement of performance measures, but also graphical illustrations of the improved real-data extracted behaviors.

### 5.1 Performance measures

To evaluate the learned weights, we use Rome subway scene. First we calculate the Silhouette index variating the cluster number between 15 and 100. The results show that a "good" cluster number for this dataset is 48. We then calculate the four performance measures (TP, FP, Dispersion, Quality) changing the cluster number around 48 (between 45 and 51 clusters). The validation is done by comparing the results of the performance measures between weighted and non weighted clustering. In the figure 8(b) *Dispersion* comparison shows that when using weights almost none of the real-data behaviors are splited (Dispersion is close to 1).
In the figure 8(a) the quality of the clusters is compared to the ground truth behaviors, showing that when using weights the distance of the RDCs to the associated GTCs is lower than with non weighted clustering. Finally in figure 8(c) the TP and FP measures are also compared.

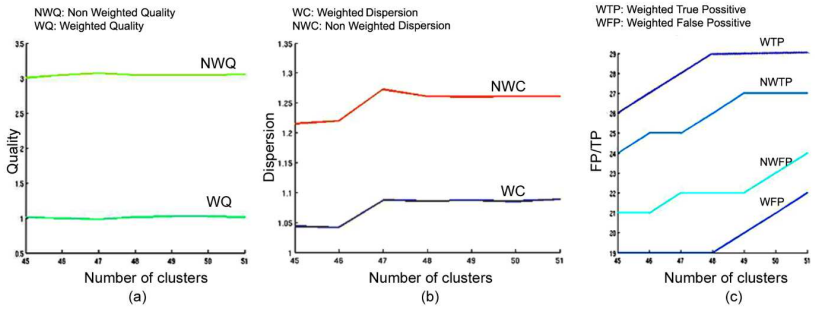


**Fig. 8.** Performance Measures Comparisons

### 5.2 Identified Activities

To better explain the improvement in clustering performance we present several illustrations between weighted and non weighted clusters. For instance in figure 9(a) and (b) the improvement of Dispersion from non weighted clusters (a) to weighted cluster (b): two clusters are present in (a) corresponding to the same behavior while in (b) only one has captured the whole amount of trajectories. The figure 8(c) shows that without weights several trajectories that describe different behaviors are merged into one cluster, the weighted clusters obtained by decreasing Dispersion allows to identify more behaviors for the same data (d). Among the identified activities by the clustering process we can enumerate: people exiting through the North Gates, people passing by the office, people stopping by the vending machine and people entering by the main entrance.

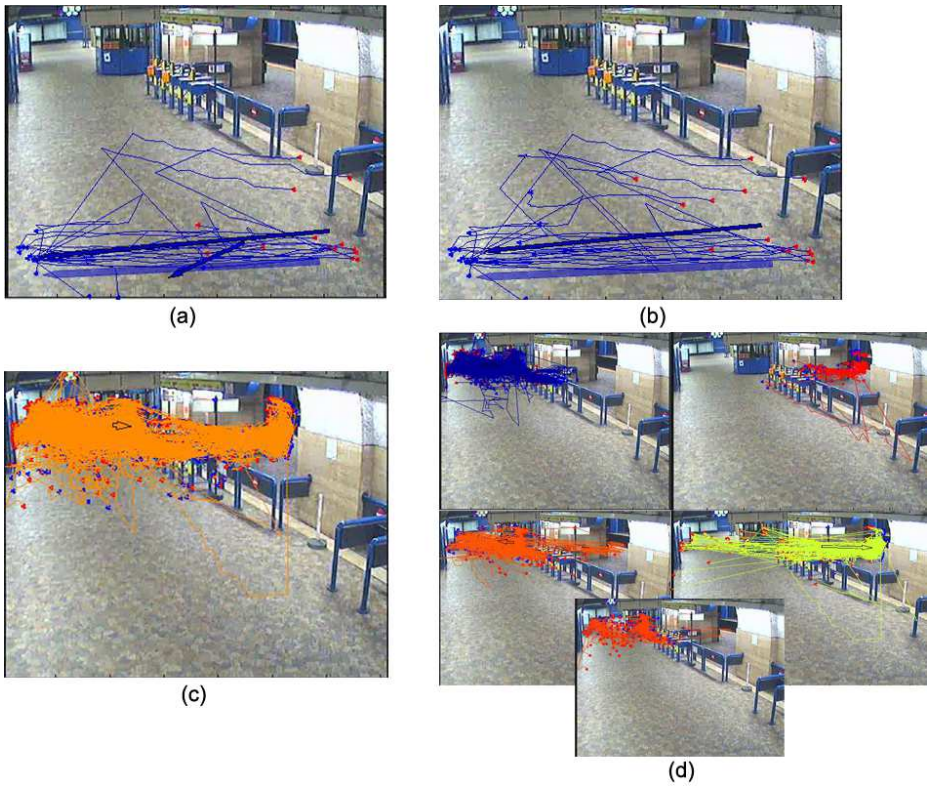


**Fig. 9.** Several identified behaviors are illustrated by weighted and non weighted clusters.

## 6 Conclusion

In this paper we propose a new approach for trajectory clustering in order to identify behaviors of people evolving in unstructured scenes (e.g., subway stations). Trajectories are represented by a simple but generic structure enabling to process any type of trajectory. We present two main contributions. First we describe how to determine the cluster number by finding the best value maximizing a clustering index criteria (i.e., Silhouette). Second we propose several performance measures to evaluate the quality of the obtained clusters. Based on these measures we apply genetic algorithm to tune the parameters of the clustering process in order to improve the overall system performance. Thus we present a complete framework for trajectory clustering and activity recognition. This framework has been tuned using training data coming from Torino subway and has been validated with testing videos taken in Rome subway. Future work includes evaluating the impact of the normalization and filtering stages to improve the cluster quality and evaluating the approach capability of detecting abnormal behaviors.

## References

1. Avanzi, A., Bremond, F., Thonnat, M.: 'Design and assessment of an intelligent activity monitoring platform', EURASIP, 2005, pp. 2395-1302
2. Moeslund, T., Hilton, A., Krger, V.: A survey of advances in vision-based human motion capture and analysis, Computer Vision and Image Understanding, 2006, 104, (2-3), pp. 90-126.
3. P.J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, J. Comp App. Math, vol. 20, pp. 53-65. 1987.
4. Zhang, H., Kantankanhalli, A., Smoliar, S.: Automatic Partitioning of Full-Motion Video, ACM Multimedia Systems, 1993, 1, (1), pp. 10-28.
5. Ewerth, R., Freisleben, B.: Semi-supervised learning for semantic video retrieval, Proc. 6th ACM int. conf. on Image and video retrieval CIVR '07, 2007.
6. Smeaton, A. F.: Techniques used and open challenges to the analysis, indexing and retrieval of digital video, Information Systems, 2007, 32, (4), pp. 545-559.
7. N. Johnson and D. Hogg, Learning the distribution of object trajectories for event recognition, in Proc. British Conf. Machine Vision, vol 2, Sept 1995, pp.583-592.
8. J. Owens and A. Hunter, Application of the self-organism map to trajectory classification, in Proc. IEEE Trans. Sys., Man, Cybern. B, vol.34 no. 3, pp 1618-1626. june 2004.
9. Le, T.-L., Boucher, A., Thonnat, M. : 'Subtrajectory-based video indexing and retrieval', Proc. 13th Int. Conf. on Advances of Multimedia Modelling, 2007, 1, pp. 418-427.
10. Patino, L. and Benhadda, H. and Corvee, E. and Bremond, F. and Thonnat, M.: 'Extraction of activity patterns on large video recordings', Computer Vision, IET, 2008, pp. 108-128.
11. H. Kargupta, The gene expression messy genetic algorithm, in Proc. IEEE Internat. Conf. on Evolutionary Computation. Nagoya University, Japan, May, 1996 (1996).
12. Piciarelli, C., Foresti, G. L., Snidaro, L.: Trajectory Clustering and its Applications for Video Surveillance, 2006.

13. Anjum, N., Cavallaro, A. : Single camera calibration for trajectory-based behavior analysis, IEEE Int. Conf. on Advanced Video and Signal Based Surveillance, AVSS '07, 2007.
14. Naftel, A., Khalid, S.: Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space, Trans. of Multimedia Systems, 2006, 12, (3), pp 45-52.
15. Antonini, G., Thiran, J. P.: Counting pedestrians in video sequences using trajectory clustering, IEEE Trans. on Circuits and Systems for Video Technology, 2006, 16, (8), pp 1008-1020.
16. Calderara, S., Cucchiara, R., Prati, A. : Detection of Abnormal Behaviors using a Mixture of Von Mises Distributions, IEEE Int. Conf. on Advanced Video and Signal Based Surveillance, AVSS '07, 2007.
17. Gaffney, S., Smyth, P.: Trajectory clustering with mixtures of regression models, Proc. Int. Conf. on Knowledge Discovery and Data Mining, 1999, California, USA.
18. Porikli, F.: Learning object trajectory patterns by spectral clustering, Proc. IEEE Int. Conf. on Multimedia and Expo ICME '04, 2004, 2, pp. 1171-1174.
19. Oliver, N.M., Rosario, B., Pentland, A.P.: A Bayesian computer vision system for modeling human interactions, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22, (8), pp. 831-843.